Contents

Request Routing

5 minute read page test

This task shows you how to route requests dynamically to multiple versions of a microservice.

Before you begin

- Setup Istio by following the instructions in the Installation guide.
- Deploy the Bookinfo sample application.

- Review the Traffic Management concepts doc. Before attempting this task, you should be familiar with important terms such as *destination rule*, *virtual service*, and *subset*.

About this task

The Istio Bookinfo sample consists of four separate microservices, each with multiple versions. Three different versions of one of the microservices, `reviews`, have been deployed and are running concurrently. To illustrate the problem this causes, access the Bookinfo app's `/productpage` in a browser and refresh several times. You'll notice that

sometimes the book review output contains star ratings and other times it does not. This is because without an explicit default service version to route to, Istio routes requests to all available versions in a round robin fashion.

The initial goal of this task is to apply rules that route all traffic to `v1` (version 1) of the microservices. Later, you will apply a rule to route traffic based on the value of an HTTP request header.

Apply a virtual service

To route to one version only, you apply virtual services that set the default version for the microservices. In this case, the virtual services will route all traffic to `v1` of each microservice.

If you haven't already applied destination rules, follow the instructions in Apply Default Destination Rules.

1. Run the following command to apply the virtual services:

```
$ kubectl apply -f @samples/bookinfo/networking/virtual-service-all-v1
.yaml@
```

Because configuration propagation is eventually consistent, wait a few seconds for the virtual services to

take effect.

2. Display the defined routes with the following command:

```
$ kubectl get virtualservices -o yaml
- apiVersion: networking.istio.io/v1beta1
  kind: VirtualService
  ...
  spec:
    hosts:
    - details
    http:
    - route:
      - destination:
          host: details
          subset: v1
- apiVersion: networking.istio.io/v1beta1
  kind: VirtualService
  ...
  spec:
    hosts:
    - productpage
```

```
    http:
    - route:
      - destination:
          host: productpage
          subset: v1
- apiVersion: networking.istio.io/v1beta1
  kind: VirtualService
  ...
  spec:
    hosts:
    - ratings
    http:
    - route:
      - destination:
          host: ratings
          subset: v1
- apiVersion: networking.istio.io/v1beta1
  kind: VirtualService
  ...
  spec:
    hosts:
    - reviews
```

```
    http:
    - route:
      - destination:
          host: reviews
          subset: v1
```

3. You can also display the corresponding `subset` definitions with the following command:

```
$ kubectl get destinationrules -o yaml
```

You have configured Istio to route to the `v1` version of the Bookinfo microservices, most importantly the `reviews` service version 1.

Test the new routing configuration

You can easily test the new configuration by once again refreshing the `/productpage` of the Bookinfo app.

1. Open the Bookinfo site in your browser. The URL is `http://$GATEWAY_URL/productpage`, where `$GATEWAY_URL` is the External IP address of the ingress, as explained in the `Bookinfo` doc.

   Notice that the reviews part of the page displays with no rating stars, no matter how many times you refresh. This is because you configured Istio to route all traffic for the

reviews service to the version `reviews:v1` and this version of the service does not access the star ratings service.

You have successfully accomplished the first part of this task: route traffic to one version of a service.

Route based on user identity

Next, you will change the route configuration so that all traffic from a specific user is routed to a specific service version. In this case, all traffic from a user named Jason will be routed to the service `reviews:v2`.

Note that Istio doesn’t have any special, built-in understanding of user identity. This example is enabled by the fact that the `productpage` service adds a custom `end-user` header to all outbound HTTP requests to the reviews service.

Remember, `reviews:v2` is the version that includes the star ratings feature.

1. Run the following command to enable user-based routing:

   ```
   $ kubectl apply -f @samples/bookinfo/networking/virtual-service-reviews-test-v2.yaml@
   ```

2. Confirm the rule is created:

```
$ kubectl get virtualservice reviews -o yaml
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
...
spec:
  hosts:
  - reviews
  http:
  - match:
    - headers:
        end-user:
          exact: jason
    route:
    - destination:
        host: reviews
        subset: v2
  - route:
    - destination:
        host: reviews
        subset: v1
```

3. On the `/productpage` of the Bookinfo app, log in as user `jason`.

   Refresh the browser. What do you see? The star ratings appear next to each review.

4. Log in as another user (pick any name you wish).

   Refresh the browser. Now the stars are gone. This is because traffic is routed to `reviews:v1` for all users except Jason.

You have successfully configured Istio to route traffic based on user identity.

Understanding what happened

In this task, you used Istio to send 100% of the traffic to the `v1` version of each of the Bookinfo services. You then set a rule to selectively send traffic to version `v2` of the `reviews` service based on a custom `end-user` header added to the request by the `productpage` service.

Note that Kubernetes services, like the Bookinfo ones used in this task, must adhere to certain restrictions to take advantage of Istio's L7 routing features. Refer to the Requirements for Pods and Services for details.

In the traffic shifting task, you will follow the same basic pattern

you learned here to configure route rules to gradually send traffic from one version of a service to another.

Cleanup

1. Remove the application virtual services:

   ```
   $ kubectl delete -f @samples/bookinfo/networking/virtual-service-all-v1.yaml@
   ```

2. If you are not planning to explore any follow-on tasks, refer to the Bookinfo cleanup instructions to shutdown the application.